DigimonGPT: An Evolvable Agent with Hierarchical Human-like Memory for Video Question Answering

Borui Li[1], Xingcai Zhang[1], Tianen Liu[1], Shuai Wang[1], Yun Cheng[2], Shuai Wang[1]*

[1]Southeast University, Nanjing, China
[2]ETH Zurich, Switzerland
libr@seu.edu.cn, zhangxingcai@seu.edu.cn, toliutianen@seu.edu.cn, shuaiwang_iot@seu.edu.cn, chengyu@ethz.ch, shuaiwang@seu.edu.cn

Abstract

Video question answering (VideoQA), whose goal is to produce answers through the integration of linguistic and visual understanding, has emerged as a significant research focus. Although Large Multimodal Models (LMMs) and autonomous agent methods have achieved notable advances in VideoQA, excessive computational overhead and restricted multimodal interaction capabilities limit their ability to facilitate the continuous evolution of the VideoQA system. To address the challenge, we introduce DigimonGPT, an evolvable VideoQA agent inspired by cognitive psychology. Specifically, DigimonGPT integrates a multimodal memory mechanism to achieve the continuous evolution of VideoQA systems. An intra-video declarative memory contains fundamental features of the video and semantic contexts extracted from historical QA pairs. Another inter-task procedural memory encodes task-solving experience for further question answering. Additionally, we introduce a hierarchical memory replay mechanism for VideoQA that selects appropriate memories by their relevance and question complexity. Extensive experiments demonstrate that DigimonGPT's accuracy outperforms by an average of 13.71% on NExT-QA datasets and 9.89% on Intent-QA datasets over LMM and autonomous agents.

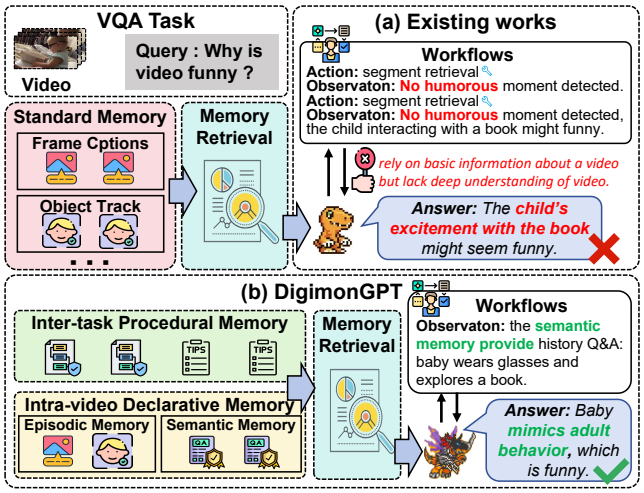

Figure 1: Comparison of existing VideoQA agents and this paper. (a) Existing works: Answer with basic information about the video. (b) DigimonGPT: Evolves with hierarchical biomimetic memories and solves the problem.

Introduction

As video emerges as the primary medium for information dissemination, Video Question Answering (VideoQA) has advanced significantly to streamline the often tedious process of extracting information from lengthy videos (Wang et al. 2024; Liu et al. 2024b; Wei et al. 2023). Prevalent Large Multimodal Models (LMMs) (Jiang et al. 2025; Chen et al. 2025; Gao et al. 2025) further accelerate the development by facilitating interaction between VideoQA systems and users through a vision-centric dialogue. Example applications include scene understanding (Azuma et al. 2022; Lin et al. 2023) and intelligent assistant (Tanaka et al. 2023).

Despite these significant advancements, current VideoQA systems still struggle to handle evolving QA tasks and diverse video content while consistently generating accurate responses. Specifically, a VideoQA system continuously learns from past tasks to deepen its understanding and accumulate experiential knowledge. This would significantly improve the quality and efficiency of its responses. To enhance this capability, existing approaches fine-tune LMMs to adapt to new tasks and video types (Feng et al. 2024; Zhao et al. 2023). While such fine-tuning can improve question-answering performance, it is limited by task distribution biases present in pretraining data (Li, Peng, and Zhou 2024; Jin et al. 2024) and incurs substantial computational overhead (D'Alessandro et al. 2023; Yu et al. 2024; Jha, Gong, and Yao 2024). To counter this problem, researchers build autonomous agents (Yang et al. 2024; Shang et al. 2024; Mao et al. 2023; Zhang et al. 2024a) and facilitate their continuous evolution. These agents leverage the reasoning and planning ability of LLMs to invoke different tools, extract necessary information, and solve the task. To enable agent evolution, recent studies (Zhang et al. 2024b; Majumder et al. 2024) depart from conventional gradient-based learning by leveraging textual memories drawn from previous tasks. These memories are both more interpretable and more efficient (Yuksekgonul et al. 2025). However, current evolution techniques of agents focus primarily on natural

*Corresponding author: Shuai Wang, shuaiwang@seu.edu.cn.

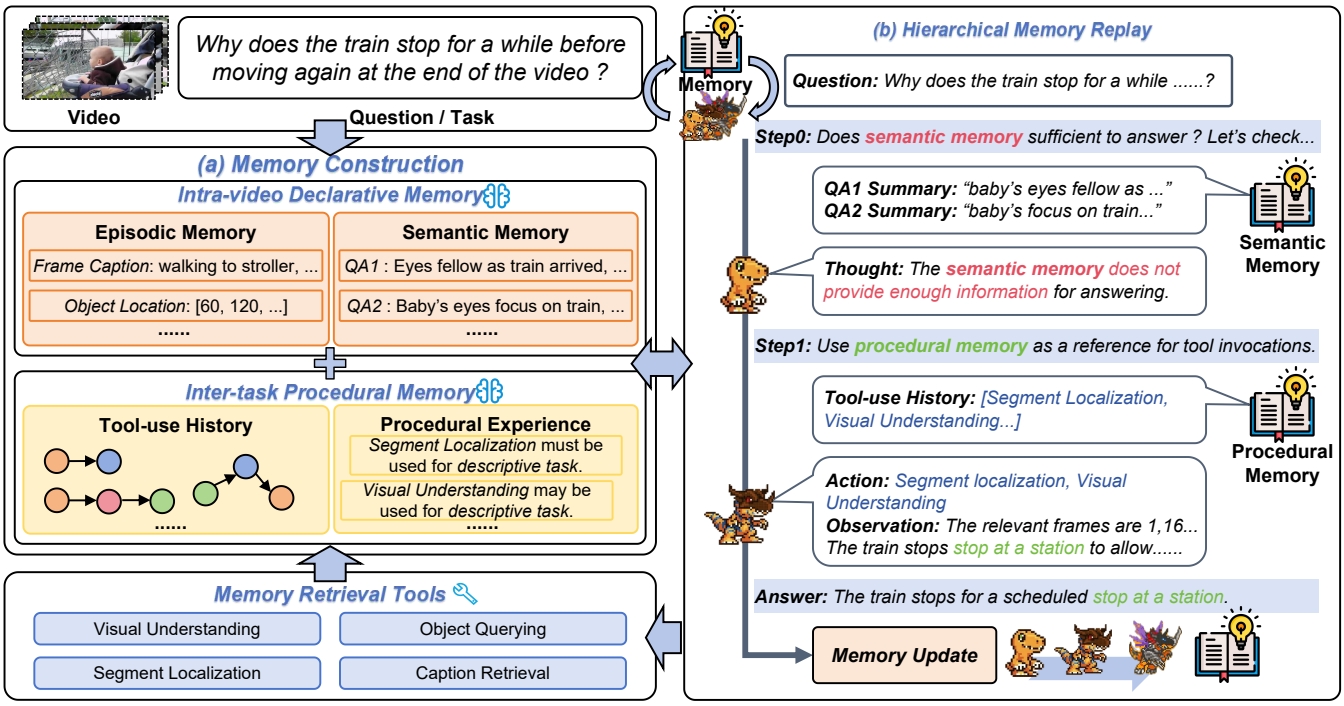

Figure 2: System architecture of DigimonGPT. Showing the integration of memory construction, memory replayable into a unified framework, and an example of DigimonGPT reasoning.

language interaction scenarios, with limited support for multimodal interactions that are essential for VideoQA systems.

Facilitating the multimodal evolution of VideoQA agents is not easy. Specifically, two non-negligible challenges arise: 1) *Designing memory architecture tailored for multimodal agent.* An evolvable VideoQA agent requires a memory architecture that organizes rich multimodal information from video-question interactions. As illustrated in Figure 1(a), the existing works (Fan et al. 2024; Yang et al. 2024) are based on structured memory, which captures surface-level visual cues but lacks deeper understanding and contextual grounding. To support sophisticated reasoning and generalization, the memory architecture should simultaneously incorporate additional information such as past task dependencies and behavioral patterns. 2) *Balancing the quantity and effectiveness of memory replay.* As knowledge accumulates, selecting and utilizing the most relevant pieces of knowledge for new tasks becomes increasingly difficult. Excessive redundant knowledge introduces semantic noise (Shi et al. 2023; Liu et al. 2024a), complicating the Q&A mechanisms from effectively identifying essential semantic cues.

Fortunately, the theory of long-term memory organization in cognitive psychology (Tulving 2002) sheds light on addressing the aforementioned challenges. In this theory, human long-term memory is categorized into *declarative memory*, which means recalling facts or concepts, and *procedural memory*, which involves recalling how to perform actions or procedures. Inspired by this, we propose DigimonGPT, an evolvable multimodal agent for VideoQA, as illustrated in Figure 1(b). Specifically, we introduce a complementary memory mechanism to retain extra task-relevant memories. First, an *Intra-video Declarative Memory* stores the declarative concepts of the video. It includes the episodic understanding of the video, as well as the key semantic information learned from past QA pairs. Second, an *Inter-task Procedural Memory* encodes the successful task-solving experience as evolving action sequences for further reuse. Based on these two kinds of memory, we propose a *Hierarchical Memory Replay* mechanism that selects appropriate memories by their relevance and question complexity, ensuring the efficient reuse of the semantic understanding and reasoning for complex VideoQA tasks.

Our main contributions are summarized as follows:

- We propose DigimonGPT, the first evolvable agent for video question answering, synergizing declarative memory (semantic retention) with procedural memory (experience evolution).
- Inspired by the memory organization theory in cognitive psychology, we propose an Intra-video Declarative Memory and an Inter-task Procedural Memory to store task-relevant features and historical retrieval experience. We further propose a Hierarchical Memory Replay mechanism that dynamically balances memory utilization to ensure accurate responses.
- We conduct extensive evaluations on representative tasks and compare DigimonGPT with VideoQA systems based on LMM and LLM-driven agents. The results demonstrate the superiority of DigimonGPT, achieving an average accuracy improvement of 13.71% on the NExT-QA dataset and 9.89% on the Intent-QA dataset.

Preliminary

LLM-based Agents. LLM-based agents are intelligent, autonomous systems designed to tackle complex tasks in dynamic environments. These tasks can be naturally framed as decision-making problems under uncertainty, where the agent cannot fully observe the true state of the environment. Instead, it must rely on partial observations and interact iteratively with the environment to achieve its objectives. The agent operates by maintaining a history of its previous actions and observations, along with a set of instructions or contextual information, such as examples, task descriptions, or environmental cues. Based on this accumulated context, the LLM functions as a policy that generates the next action to take. Each action leads to a new response from the environment, including a new observation and potentially a change in state. This process continues step by step, with the agent using its updated history to inform subsequent actions. The interaction ends when the task is successfully completed or when a predefined step limit is reached.

Working-memory Buffer. From the cognitive science perspective, working memory enables individuals to hold and manipulate information in real-time, facilitating complex cognitive tasks such as reasoning, comprehension, and learning (Barsalou 2014). In DigimonGPT , memory refers to the critical history the model needs at any given moment to perform its current task effectively. Managing this memory well enables the agent to integrate past experiences with present inputs, leading to more accurate and context-aware decisions. This process mirrors human attentional control and cognitive updating, where relevant information is prioritized, distractions are filtered out, and new inputs are continuously incorporated into a mental workspace. A common approach, shown in Figure 1, stores the entire history of interactions in memory. While this ensures the model has complete context, it can also introduce unnecessary redundancy and increase the cognitive load on the LLM.

Method

Overview

The core idea of DigimonGPT is to employ biomimetic memory that enables structured storage and dynamic retrieval of multimodal video and interaction information, enhancing the agent's capabilities in semantic understanding. More specifically, as shown in Figure 2. DigimonGPT consists of two interconnected components: memory construction and hierarchical memory replay. Based on the declarative and procedural memories, DigimonGPT utilizes the hierarchical memory replay mechanism by selecting the appropriate type and amount of memories that help answer the current question. Furthermore, at the end of each question-answering task, DigimonGPT automatically updates the two kinds of memory to facilitate the evolution of the agent.

Problem Formulation

We consider DigimonGPT is built on a large language model L and a text-based composite memory M, and we categorize its memory into declarative memory within videos and procedural memory across tasks.

Instruction 1: Verify the logic of historical Q&A.

Task description: Your task is to evaluate whether the answer provided by the agent for a video QA task logically answers the user's question.
Please based on the following two components:
1) Logical Check: Does the answer follow the question and make sense in context?
2) Semantic Check: Is the answer meaningful and accurate in terms of the question's intent?
question:{question}
answer: {answer}

(a) Validation Instruction

Instruction 2: Summarize historical Q&A.

Task description: Your task is to summarize the following questions and answers. **Please based on the following two components:**
1) Ensure that no details related to the video are overlooked in the question and answer.
2) Summarize the content as concisely as possible while retaining all key information.
question:{question}
answer: {answer}

(b) Summary Instruction

Instruction 3: Distill tool usage workflow.

Task description: Your task is to extracting the decision-making patterns of the tasks based on the successful workflow of tool invocations and generating a general operation guide for similar tasks in the future.
Please pay attention to the following points:
1) Identify the key tools required for each task type, e.g., "xx must be used for xx task. xx may be used for xx task. "
2) The dependencies and sequence of tool invocations. The key decision points that may arise during the task execution.
Question category: {question_category}
Workflow: {workflow}

(c) Workflow Distillation Instruction

Figure 3: Memory Construction Instructions

At the beginning of a task, we assume that the agent has no memory. In order to complete the task specified by the user's query q and video v, the agent first needs to preprocess the video, converting it into structured episodic memory M_e. Then, at each time step t_i, the environment state s_i produces an observation o_i, which is passed to the model L. The model computes an action $a_i = L(q, M_e, o_i)$, which is then executed in the environment. This action is executed in the environment, changing the state to $T(s_i, a_i) \rightarrow s_{i+1}$, where T represents the environment state update function. This observation-action loop continues until the model predicts a stopping action a_i = STOP, or a task termination condition is met, such as reaching the maximum allowed steps or the model deriving an answer.

Each completed task forms an experience e, which includes the user's instruction q, the agent's answer a, and the sequence of steps taken during the task-solving process. Our goal is to build a set of experiences $E = e$ from these completed tasks, and infer semantic memory M_s and procedural memory M_p. Ultimately, these inferred memories are added to the agent's memory store M, and hierarchical memory replay is used to guide the solving of subsequent tasks.

Memory Construction

In this section, we will provide a detailed explanation of how different types of memories are constructed and the rationale behind each memory type.

Intra-video Declarative Memory. To enhance contextual understanding and reasoning in VideoQA, we propose Intra-video Declarative Memory, inspired by the cognitive distinction between episodic and semantic memory. This

framework allows DigimonGPT to store both detailed video observations and abstract semantic knowledge, forming a comprehensive understanding of the video content.

(1) *Episodic Memory via Video Feature Extraction.* Our episodic memory M_e is used to store captions and related features of video segments. For each video segment, we extract captions, video features, and text embeddings. Additionally, objects in the video (such as people and items) are tracked, and their frame numbers and locations are recorded, along with the features of these objects for later retrieval. By capturing key information from the video, DigimonGPT is able to gain an initial understanding of the video content. The specific models used for video preprocessing will be discussed in the deployment details section.

(2) *Semantic Memory via Past QA Summarization.* The purpose of constructing semantic memory M_s is to store high-quality historical QA, helping agents develop a deep understanding of the video content. By dynamically updating the semantic memory, DigimonGPT can more efficiently understand and respond to user questions about the deeper meaning of the video, while avoiding redundant tool usage. Specifically, our semantic memory stores historical QA pairs validated by a large language model (LLM). Each pair is verified to ensure its correctness and semantic consistency and is then transformed into a summary without redundant information (validation and summary prompts are shown in Figure 3). We then use a text encoder to convert these summaries into semantic vectors and store them in a dedicated vector database D_s.

Inter-task Procedural Memory. Procedural memory M_p records the workflows during task execution, specifically the sequence of tool invocations used by the agent throughout the task. This enables the agent to handle similar tasks more efficiently (e.g., if the agent has encountered similar questions in different videos before, and the solutions to those questions are consistent). Additionally, by abstractly summarizing the decision patterns for various task types, we enhance the agent's ability to reason across tasks and mitigate the negative impact of low-quality workflows. Our procedural memory is designed to store the tool invocation workflows of completed tasks, with each workflow representing the sequence and dependencies of tool invocations. As tasks progress, the system continuously archives and stores successful workflows for different task types in the vector database D_p. To improve the memory's applicability and generalization, we introduce a task-type abstraction mechanism, categorizing tasks into causal, temporal, and descriptive types (Xiao et al. 2021). Once the number of completed tasks in each category exceeds five, the system uses the prompt template shown in Figure 3 to summarize and generate general decision patterns through an LLM.

Hierarchical Memory Replay

Upon completion of memory construction, we introduce a hierarchical memory replay mechanism to enable DigimonGPT to perform question answering tasks with both efficiency and precision. In the following sections, we describe the memory retrieval process in detail and explain how the system dynamically adjusts the replay quantity of different

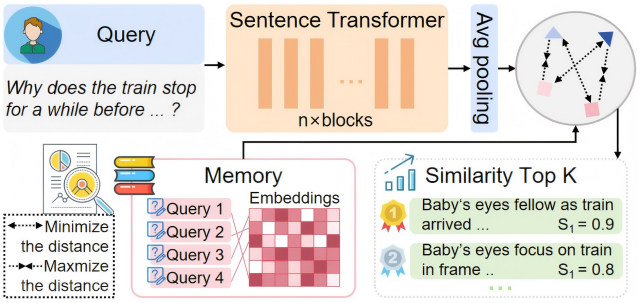

Figure 4: Memory retrieval mechanism of DigimonGPT.

memory types based on task complexity.

Memory Retrieval. The method aims to accurately identify the most relevant knowledge to the current user query. DigimonGPT achieves this by retrieving the required memory from three different vector databases. For the sake of explaining the functionality of the components, we use the retrieval of semantic memory M_s as an example to illustrate the specific memory retrieval process. As shown in Figure 4, the method implements a retrieval function $R : q \times M_s \rightarrow M_s^{(s)}$, where q represents the query space, M_s is the memory store, and $M_s^{(s)} \subset M_s$ is the subset of memory entries deemed relevant to the given query. This retrieval process is crucial for effective memory reuse, as it filters out only the information that is relevant to the current task from the memory store, enabling precise memory application.

The core of memory retrieval relies on semantic similarity measurement between the current user query and relevant questions in the memory store. Retrieval based on similarity can be formally defined as:

$$R(q, M_s) = \{m_j \in M_s \mid \text{sim}\,(q, q_j) \geq \tau_s\}_{j=1}^{\text{top-k}} \quad (1)$$

Here, q is the current query, q_j is the question related to memory entry m_j, $\text{sim}\,(,)$ is the similarity function, τ_s is the similarity threshold (τ_s=0.5 by default), and top-k refers to selecting the most similar entries of k. To achieve this similarity measurement, we adopt a two-stage processing approach:

1. Embedding Generation: Convert the user query into a dense vector representation. This is achieved using the text-embedding-3-large model via remote invocation.
2. Similarity Calculation: Compute the cosine similarity between the embedding vectors to quantify the semantic relationship between queries.

Through this approach, we can efficiently filter out the most relevant memory entries from a large set, enabling precise memory reuse tailored to the current query.

Complexity-Aware Memory Adjustment. We observe that excessive memory can impair inference performance, while insufficient memory fails to enhance reasoning capabilities (Shi et al. 2023; Liu et al. 2024a). We dynamically adjust the number of memory replays to adapt to task complexity. This mechanism intelligently retrieves an optimal amount of memory input, ensuring that the model receives adequate support without being overloaded.

Instruction 4: Assess semantic complexity of the question.
Task description: Please rate each question on a scale from 0 to 1 (with 1 being the highest complexity), considering the depth of reasoning and contextual understanding required. Questions that are more detailed or involve more objects require a more complex answer process.
question:{question}

Figure 5: Instruction of semantic complexity assessment.

First, we employ a large language model (LLM) to provide a prompt-based assessment of the semantic complexity of question q, as shown in figure 5. The LLM evaluates the question based on factors like syntactic structure, reasoning demands, and contextual dependencies, and assigns a normalized score $P(q) \in [0, 1]$ that reflects its semantic complexity. Next, we introduce keyword entropy $E\left(Keywords\left(\cdot\right)\right)$ to capture the diversity and richness of the query's semantic content. We begin by tokenizing the input text and removing common stopwords using the NLTK English stopword list. After filtering, we compute the frequency of the remaining keywords and normalize these counts to obtain a probability distribution. Finally, we calculate the Shannon entropy of this distribution to quantify the semantic diversity and informational richness of the question. Finally, to refine the overall complexity score $C\left(\cdot\right)$, we employ Bayesian optimization (Shahriari et al. 2015) to automatically determine the optimal values for hyperparameters α and β. Therefore, the overall complexity of a given query q is depicted in Eq. 2.

$$C\left(q\right) = \alpha \cdot P\left(q\right) + \beta \cdot E\left(Keywords\left(q\right)\right) \quad (2)$$

Based on $C\left(q\right)$, the system dynamically adjusts the number of items replayed from semantic memory M_s and episodic memory M_p. Specifically, the number of M_s and M_p replays, denoted as k_s and k_p, is determined by Eq. 3 and 4. In addition, we predefine the minimum and maximum values of k_s and k_p as 1 and 5, respectively, resulting in the repaly ranges $[k_{s,\text{min}}, k_{s,\text{max}}]$ and $[k_{p,\text{min}}, k_{p,\text{max}}]$.

$$k_s = round\left(k_{s,min} + C\left(q\right) \cdot \left(k_{s,max} - k_{s,min}\right)\right) \quad (3)$$

$$k_p = round\left(k_{p,min} + C\left(q\right) \cdot \left(k_{p,max} - k_{p,min}\right)\right) \quad (4)$$

We employ the rounding function $round(\cdot)$ to convert the intermediate continuous values into discrete integers. By dynamically adjusting the replay size in this manner, the system optimizes performance and provides sufficient information for complex queries while avoiding unnecessary overload for simpler ones.

Experiments

Experimental Setup

Implementation Details. We develop the system using PyTorch and LangChain (Topsakal and Akinci 2023), integrating GPT-4o-mini (Roumeliotis and Tselikas 2023) for language processing. The preprocessing pipeline employs LaViLa (Zhao et al. 2023) for frame-level captioning, followed by ViCLIP (Wang et al. 2023b) and text-embedding-3-large (Wang et al. 2023a) for visual-textual feature embedding. Spatiotemporal analysis combines RT-DETR (Zhao et al. 2024) for object detection and ByteTrack (Zhang et al. 2022) for multi-object tracking. Experiments on an NVIDIA RTX 3090 GPU adopt in-context learning, with frozen parameters to ensure fair comparisons.

Datasets. We utilize two datasets for our experiments: NExT-QA (Xiao et al. 2021) and Intent-QA (Li et al. 2023b). The NExT-QA dataset comprises 34,132 video-question pairs for training and 4,996 pairs for validation. Each question is labeled with one of three types: Causal (Cau.), Temporal (Tem.), or Descriptive (Des.), which is accompanied by five candidate answers. The Intent-QA dataset focuses on intent reasoning in social scenarios, comprising 4.3k videos and 16k pairs, question types of Causal-Why (CW.), Causal-How (CH.), and Temporal (Tem.). During evaluation, the test samples are randomly selected and shuffled, and the system's memory is reset before each evaluation round.

Metrics. We evaluate DigimonGPT using the different accuracy metrics. On the NExT-QA dataset, we report accuracy for the above three question types, as well as the overall accuracy across all questions (All). For the Intent-QA dataset, the evaluation method of DigimonGPT is similar to that of the NExT-QA dataset.

Baseline. We compare DigimonGPT with VideoQA systems based on MM (Multimodal Mode), LMM, and LLM-driven agents. *MM:* HGA (Jiang and Han 2020) utilizes a heterogeneous graph alignment network to achieve synchronized alignment and interaction across and within modalities. *LMM:* Video-LLaVA (Lin et al. 2024) unifies image and video representations, allowing the base large language model to handle multimodal tasks more effectively. *LLM-driven agents:* LangRepo (Kahatapitiya et al. 2025) updates video captions to assist LLMs in video QA tasks, we use a proprietary LLM for this experiment. VideoAgent (Fan et al. 2024) constructs a structured memory to store general event descriptions and object-based state tracking information extracted from videos. TraveLER (Shang et al. 2024) employs a modular agent architecture with a feedback mechanism, enabling more flexible reasoning strategies.

Main Results

Table 1 presents a performance that DigimonGPT outperforms state-of-the-art methods on both the NExT-QA and Intent-QA datasets. Our system outperforms the traditional model (HGA) by 16.86%, LMM (Video-LLaVa) by 12.93%, and the LLM-driven agent (VideoAgent & TraveLER) by 9.27% in terms of average accuracy across the two datasets.

Specifically, on the NExT-QA dataset, since HGA (Jiang and Han 2020) is constrained by task distribution biases inherent in its pretraining data, and Video-LLaVa (Lin et al. 2024) analyzes only a limited number of video frames to generate responses, their overall performance is 19.11% and 14.61% lower, respectively, compared to our system. In addition, DigimonGPT improves the overall accuracy by 8.99% compared to VideoAgent (Fan et al. 2024). This enhancement stems from incorporating supplemental semantic and procedural memory, which enables systematic replay of historical memories during reasoning, yielding more precise responses. While TraveLER (Shang et al. 2024) demonstrates second-best overall performance, it still lags with our

Dataset	NExT-QA				Intent-QA			
Question Type	Causal	Temporal	Descriptive	All	Causal-Why	Causal-How	Temporal	All
HGA (Jiang and Han 2020)	50.00	51.72	37.5	48.31	43.33	55.17	43.33	47.19
Video-LLaVa (Lin et al. 2024)	47.73	58.62	56.25	52.81	53.33	48.28	50.00	50.56
LangRepo (Kahatapitiya et al. 2025)	59.09	44.83	56.25	53.93	60.00	**62.07**	40.00	53.93
VideoAgent (Fan et al. 2024)	56.82	55.17	50.00	55.06	56.67	58.62	43.33	52.81
TraveLER (Shang et al. 2024)	63.64	58.62	43.75	58.43	56.67	51.72	56.67	55.06
DigimonGPT (Ours)	**70.45**	**62.07**	**68.75**	**67.42**	**63.33**	58.62	**63.33**	**61.80**

Table 1: Accuracy of different question types on the NExT-QA and Intent-QA datasets.

SM	EM	PM	Causal	Temporal	Descriptive	All
	✓		56.82	55.17	50.00	55.06
✓	✓		68.18	58.62	43.75	60.67
	✓	✓	52.27	62.07	68.75	58.43
✓	✓	✓	**70.45**	**62.07**	**68.75**	**67.42**

Table 2: Accuracy analysis of DigimonGPT with different memory components.

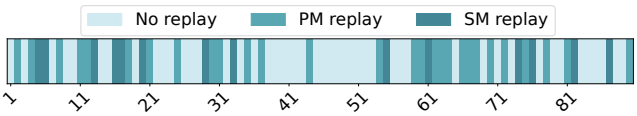

Figure 6: The semantic memory (SM) and procedural memory (PM) replay in DigimonGPT during task execution.

method by 8.99%. Especially in the descriptive task, DigimonGPT improves the accuracy by 18.75% compared with TraveLER. This is because descriptive questions necessitate a detailed understanding and reconstruction of video content. DigimonGPT addresses this by retrieving highly relevant procedural memories, which provide tool usage experiences from past successful tasks.

On the Intent-QA dataset, DigimonGPT achieves an average improvement of 9.89% over the baseline models. Specifically, DigimonGPT outperforms other methods on Causal-Why questions, surpassing VideoAgent and TraveLER by 6.66%. In temporal reasoning tasks, DigimonGPT achieves 20% improvement over VideoAgent, which also incorporates the memory mechanism. Advantage primarily arises from the strong temporal dependencies inherent in the Intent-QA dataset, where our semantic memory module plays a key role by effectively preserving and retrieving relevant contextual information. This enables the QA system to make more informed and accurate inferences.

Ablation Studies

To gain deeper insights into the performance and effectiveness of DigimonGPT, we conduct a comprehensive set of ablation experiments on the NExT-QA dataset.

Memory Components. We investigate the essential memory components in DigimonGPT, i.e., semantic memory (SM) and episodic memory (EM) within declarative memory, and procedural memory (PM). The results are summarized in Table 2. Row 1 vs. Row 2: The addition of Semantic Memory (SM) in Row 2 leads to a significant improvement in Causal reasoning accuracy, rising from 56.82% to 68.18%. This confirms that SM plays a vital role in capturing inter-segment causal dependencies in tasks, which are difficult to infer based solely on episodic memory. Interestingly, we observe moderate gains in the Temporal task (from 55.17% to 58.62%), but a drop in Descriptive task accuracy (from 50.00% to 43.75%). This fluctuation is attributed to the lack of Procedural Memory (PM): without PM, the agent depends solely on the LLM to select tools, which can lead to inconsistent behavior and limited fine-grained understanding, especially for descriptive questions that require detailed video reconstruction. Row 3 vs. Row 4: Both configurations include Procedural Memory (PM), which helps stabilize performance in Temporal and Descriptive tasks. Specifically, accuracy on the Descriptive task rises sharply to 68.75%, verifying the effectiveness of PM in storing successful tool-use trajectories and mitigating LLM uncertainty during action selection. However, Row 3 lacks SM and thus struggles with Causal questions, scoring 52.27%, which is 18.18% lower than Row 4. Overall, the best performance (67.42% accuracy across all tasks) is achieved when integrating all three memory components (SM + EM + PM), validating the effectiveness of our cognitively inspired memory architecture in video question answering. As visualized in Figure 6, the system replays relevant memory in up to 47.8% of tasks, demonstrating frequent memory reuse.

The Number of Memory Replay. We evaluate the impact of varying semantic memory M_s and procedural memory M_p to evaluate their efficiency in supporting LLM reasoning, as shown in Table 3. First, the overall accuracy performance of the no-replay baseline ($M_s = 0$, $M_p = 0$) is significantly lower than our full model, with a gap of 12.36%, demonstrating the critical role of memory replay in supporting reasoning. In the small-scale replay setting ($M_s = 1$, $M_p = 1$), the semantic memory module provides only the single most relevant memory item, which helps the LLM focus on precise cues during inference. Interestingly, this configuration slightly outperforms our full model by 2.28% on the causal task. However, it falls short in descriptive tasks, lagging behind by 25%. While Row 2 does include procedural memory, its replay scale ($M_p = 1$) is too limited to provide effective support for descriptive tasks. Such tasks often require detailed visual context and multi-step reasoning, which a single procedural memory entry can limit the decision-making of LLM. As a result, the LLM lacks suffi-

M_s	M_p	Causal	Temporal	Descriptive	All
0	0	56.82	55.17	50	55.06
1	1	**72.73**	55.17	43.75	61.80
5	5	61.36	51.72	56.25	57.30
3	2	68.18	58.62	68.75	65.17
DigimonGPT		70.45	**62.07**	**68.75**	**67.42**

Table 3: Accuracy of DigimonGPT compared with different numbers of memory replayed for VideoQA.

α	β	Causal	Temporal	Descriptive	All
0.11	0.89	59.09	51.72	68.75	58.43
0.03	0.3	65.91	55.17	68.75	62.92
0.05	0.2	**72.73**	58.62	62.5	66.29
0.03	**0.2**	70.45	**62.07**	**68.75**	**67.42**

Table 4: Accuracy of DigimonGPT under different hyperparameter settings.

cient reference to guide tool usage, leading to inconsistent or suboptimal decisions. In contrast, Row 1 (without procedural memory) benefits from more stable default behavior, yielding slightly better descriptive performance, which is an unpredictable and random action. Therefore, Row 2 performing worse than Row 1 on the Descriptive task is expected. It highlights that overly limited procedural replay may mislead the model more than it helps. In contrast, the large-scale replay setting (M_s = 5, M_p = 5) introduces excessive memory entries, which may introduce noise and distract the model from the most relevant information, thus affecting performance. Although M_s = 3, M_p = 2 are the best-performing parameters derived from extensive experiments, the dynamic retrieval strategy ultimately outperforms by 2.25% in overall accuracy.

Hyperparameter Analysis. We use Bayesian optimization with the HuggingFace problem complexity dataset to define the objective function (Shahriari et al. 2015). This process identifies the optimal hyperparameter configuration: $\alpha = 0.03$ and $\beta = 0.2$. To validate this configuration, we perform additional evaluations by (1) testing a normalized version of the optimal parameters and (2) fixing one parameter (α or β) while scaling the other upward.

As shown in Table 4, the configuration $\alpha = 0.03$ and $\beta = 0.2$ achieves the highest overall accuracy, outperforming the normalized version of the best hyperparameters by 8.99%. Moreover, increasing either parameter beyond the optimal values does not improve performance. Specifically, increasing β to 0.3 results in a 4.5% drop in accuracy, while increasing α to 0.05 leads to a 1.13% drop in accuracy. These results confirm the robustness of our hyperparameter configuration. DigimonGPT with $\alpha = 0.05$ and $\beta = 0.2$ achieves 2.28% higher accuracy on causal questions compared to DigimonGPT employed Bayesian optimization. Since the prompts we designed for evaluation are intended to guide the model to consider potential causal relationships and semantic associations, an increase in the value of α results in more precise evaluation scores for causal tasks. Consequently, this yields a more appropriate number of memory replay entries for these tasks and ultimately enhances performance in this specific task category.

Related Work

LMM Fine-tuning. In recent years, LMMs have achieved substantial advancements in VideoQA through unified multimodal architectures that integrate heterogeneous modalities and utilize large-scale multimodal pre-training. However, pre-trained LMMs often encounter difficulties in generalizing to novel tasks, producing unsatisfactory responses. To address these limitations, one approach involves task-adaptive fine-tuning through parameter updates (Cao et al. 2024; Feng et al. 2024; Yan et al. 2024; Jin et al. 2024), which optimizes model parameters directly on target domain data. Another strategy (Sanh et al. 2021; Li et al. 2023a) involves fine-tuning pretrained LMMs on a collection of multimodal instruction-following data, guiding LMMs to comprehend and adapt to target tasks, thereby enhancing task-specific performance. Nonetheless, fine-tuning LMMs incurs substantial computational overhead and time costs, which significantly increase system response latency.

LLM-driven Agents. Considering computational expenses and task-specific constraints, alternative approaches employ LLM-driven agents to orchestrate a multitude of models from external toolkits. ViperGPT (Surís, Menon, and Vondrick 2023) combines video and language models to generate executable subprograms for open-domain visual query parsing. TraveLER (Shang et al. 2024) progresses this framework with a multi-agent system characterized by flexible planning and reflection for VideoQA tasks, while VideoAgent and DoraemonGPT (Fan et al. 2024; Yang et al. 2024) models structured video memory via tool-based reasoning. However, existing works primarily enhance reasoning in single-task scenarios, failing to consolidate completed tasks as experiential knowledge for future tasks. This isolated approach necessitates the reinstantiation reasoning chains for each task execution, proving inefficient when managing multimodal interactions. Recent works (Yang et al. 2025; Ning et al. 2025; Di et al. 2025) have explored efficient KV-cache retrieval mechanisms for streaming video question answering, but these typically rely on high task similarity, whereas our DigimonGPT leverages hierarchical memory replay to generalize across unseen tasks with evolvable reasoning capability.

Conclusion

In this paper, we present DigimonGPT, an evolvable multimodal agent that integrates an Intra-video Declarative Memory and an Inter-task Procedural Memory to mimic human memory and knowledge transfer. By dynamically adjusting memory utilization based on evaluating question complexity, DigimonGPT enables evolving strategy without the heavy computational burden of fine-tuning approaches. We conduct extensive evaluations of DigimonGPT on multiple types of video question answering tasks and compare against LMM and LLM-driven agent VideoQA systems. Results demonstrate the advantages of DigimonGPT 's accuracy on averaged increasing 13.71% on NExT-QA datasets and 9.89% on Intent-QA datasets over baselines.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62302096, Grant U24B20152, and Grant 62272098; the Natural Science Foundation of Jiangsu Province under Grant BK20230813; the Zhishan Young Scholar Program of Southeast University under Grant 3209002402A2; and the Major Project of Fundamental Research on Frontier Leading Technology of Jiangsu Province under Grant BK20222006.

References

Azuma, D.; Miyanishi, T.; Kurita, S.; and Kawanabe, M. 2022. Scanqa: 3d question answering for spatial scene understanding. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 19129–19139.

Barsalou, L. W. 2014. *Cognitive psychology: An overview for cognitive scientists*. Psychology Press.

Cao, C.; Zhang, Y.; Yu, Y.; Lv, Q.; Min, L.; and Zhang, Y. 2024. Task-Adapter: Task-specific Adaptation of Image Models for Few-shot Action Recognition. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 9038–9047.

Chen, S.; Luo, Y.; Ma, Y.; Qiao, Y.; and Wang, Y. 2025. H-mba: Hierarchical mamba adaptation for multi-modal video understanding in autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 2212–2220.

D'Alessandro, M.; Alonso, A.; Calabrés, E.; and Galar, M. 2023. Multimodal parameter-efficient few-shot class incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3393–3403.

Di, S.; Yu, Z.; Zhang, G.; Li, H.; Cheng, H.; Li, B.; He, W.; Shu, F.; Jiang, H.; et al. 2025. Streaming Video Question-Answering with In-context Video KV-Cache Retrieval. In *The Thirteenth International Conference on Learning Representations*.

Fan, Y.; Ma, X.; Wu, R.; Du, Y.; Li, J.; Gao, Z.; and Li, Q. 2024. Videoagent: A memory-augmented multimodal agent for video understanding. In *European Conference on Computer Vision*, 75–92. Springer.

Feng, Y.; Tian, Z.; Zhu, Y.; Han, Z.; Luo, H.; Zhang, G.; and Song, M. 2024. CP-Prompt: Composition-Based Cross-modal Prompting for Domain-Incremental Continual Learning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, 2729–2738. New York, NY, USA: Association for Computing Machinery. ISBN 9798400706868.

Gao, M.; Liu, J.; Li, M.; Xie, J.; Liu, Q.; Zhao, K.; Chen, X.; and Xiong, H. 2025. TC-LLaVA: Rethinking the Transfer of LLava from Image to Video Understanding with Temporal Considerations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 3086–3094.

Jha, S.; Gong, D.; and Yao, L. 2024. Clap4clip: Continual learning with probabilistic finetuning for vision-language models. *Advances in Neural Information Processing Systems*.

Jiang, H.; Jin, Y.; Sun, Z.; Xu, K.; Chen, L.; Song, Y.; Gai, K.; and Mu, Y. 2025. Granularity-Adaptive Spatial Evidence Tokenization for Video Question Answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 3976–3984.

Jiang, P.; and Han, Y. 2020. Reasoning with heterogeneous graph alignment for video question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 11109–11116.

Jin, T.; Yan, W.; Wang, Y.; Cai, S.; Shuai, Q.; and Zhao, Z. 2024. Calibrating Prompt from History for Continual Vision-Language Retrieval and Grounding. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 4302–4311.

Kahatapitiya, K.; Ranasinghe, K.; Park, J.; and Ryoo, M. S. 2025. Language repository for long video understanding. In *Findings of the Association for Computational Linguistics ACL 2025*.

Li, J.; Li, D.; Savarese, S.; and Hoi, S. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, 19730–19742. PMLR.

Li, J.; Wei, P.; Han, W.; and Fan, L. 2023b. Intentqa: Context-aware video intent reasoning. In *Proceedings of the IEEE/CVF international conference on computer vision*, 11963–11974.

Li, Q.; Peng, Y.; and Zhou, J. 2024. Progressive Prototype Evolving for Dual-Forgetting Mitigation in Non-Exemplar Online Continual Learning. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 2477–2486.

Lin, B.; Ye, Y.; Zhu, B.; Cui, J.; Ning, M.; Jin, P.; and Yuan, L. 2024. Video-LLaVA: Learning United Visual Representation by Alignment Before Projection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 5971–5984.

Lin, H.; Zala, A.; Cho, J.; and Bansal, M. 2023. Videodirectorgpt: Consistent multi-scene video generation via llm-guided planning. *arXiv preprint arXiv:2309.15091*.

Liu, N. F.; Lin, K.; Hewitt, J.; Paranjape, A.; Bevilacqua, M.; Petroni, F.; and Liang, P. 2024a. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12: 157–173.

Liu, X.; Wan, J.; Zong, L.; and Xu, B. 2024b. Conditional Diffusion Model for Open-ended Video Question Answering. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 9057–9066.

Majumder, B. P.; Dalvi Mishra, B.; Jansen, P.; Tafjord, O.; Tandon, N.; Zhang, L.; Callison-Burch, C.; Clark, P.; Patel, A.; Raffel, C.; et al. 2024. CLIN: A Continually Learning Language Agent for Rapid Task Adaptation and Generalization. In *Conference on Language Modeling ({COLM})*. IEEE/CVF.

Mao, J.; Ye, J.; Qian, Y.; Pavone, M.; and Wang, Y. 2023. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*.

Ning, Z.; Liu, G.; Jin, Q.; Ding, W.; Guo, M.; and Zhao, J. 2025. LiveVLM: Efficient Online Video Understanding via Streaming-Oriented KV Cache and Retrieval. *arXiv preprint arXiv:2505.15269*.

Roumeliotis, K. I.; and Tselikas, N. D. 2023. Chatgpt and open-ai models: A preliminary review. *Future Internet*, 15(6): 192.

Sanh, V.; Webson, A.; Raffel, C.; Bach, S. H.; Sutawika, L.; Alyafeai, Z.; Chaffin, A.; Stiegler, A.; Scao, T. L.; Raja, A.; et al. 2021. Multitask prompted training enables zero-shot task generalization. *Proceedings of the Tenth International Conference on Learning Representations (ICLR)*.

Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; and De Freitas, N. 2015. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175.

Shang, C.; You, A.; Subramanian, S.; Darrell, T.; and Herzig, R. 2024. TraveLER: A Modular Multi-LMM Agent Framework for Video Question-Answering. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 9740–9766. Miami, Florida, USA: Association for Computational Linguistics.

Shi, F.; Chen, X.; Misra, K.; Scales, N.; Dohan, D.; Chi, E. H.; Schärli, N.; and Zhou, D. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, 31210–31227. PMLR.

Surís, D.; Menon, S.; and Vondrick, C. 2023. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11888–11898.

Tanaka, R.; Nishida, K.; Nishida, K.; Hasegawa, T.; Saito, I.; and Saito, K. 2023. Slidevqa: A dataset for document visual question answering on multiple images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 13636–13645.

Topsakal, O.; and Akinci, T. C. 2023. Creating large language model applications utilizing langchain: A primer on developing llm apps fast. In *International Conference on Applied Engineering and Natural Sciences*, volume 1, 1050–1056.

Tulving, E. 2002. Episodic memory: From mind to brain. *Annual review of psychology*, 53(1): 1–25.

Wang, H.; Lai, C.; Sun, Y.; and Ge, W. 2024. Weakly supervised gaussian contrastive grounding with large multimodal models for video question answering. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 5289–5298.

Wang, L.; Yang, N.; Huang, X.; Yang, L.; Majumder, R.; and Wei, F. 2023a. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Wang, Y.; He, Y.; Li, Y.; Li, K.; Yu, J.; Ma, X.; Li, X.; Chen, G.; Chen, X.; Wang, Y.; et al. 2023b. Internvid: A large-scale video-text dataset for multimodal understanding and generation. *arXiv preprint arXiv:2307.06942*.

Wei, Y.; Liu, Y.; Yan, H.; Li, G.; and Lin, L. 2023. Visual causal scene refinement for video question answering. In *Proceedings of the 31st ACM International Conference on Multimedia*, 377–386.

Xiao, J.; Shang, X.; Yao, A.; and Chua, T.-S. 2021. NExT-QA: Next Phase of Question-Answering to Explaining Temporal Actions. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9772–9781.

Yan, W.; Wang, Y.; Lin, W.; Guo, Z.; Zhao, Z.; and Jin, T. 2024. Low-rank Prompt Interaction for Continual Vision-Language Retrieval. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 8257–8266.

Yang, Y.; Zhao, Z.; Shukla, S. N.; Singh, A.; Mishra, S. K.; Zhang, L.; and Ren, M. 2025. Streammem: Query-agnostic kv cache memory for streaming video understanding. *arXiv preprint arXiv:2508.15717*.

Yang, Z.; Chen, G.; Li, X.; Wang, W.; and Yang, Y. 2024. DoraemonGPT: toward understanding dynamic scenes with large language models (exemplified as a video agent). In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Yu, J.; Zhuge, Y.; Zhang, L.; Hu, P.; Wang, D.; Lu, H.; and He, Y. 2024. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 23219–23230.

Yuksekgonul, M.; Bianchi, F.; Boen, J.; Liu, S.; Lu, P.; Huang, Z.; Guestrin, C.; and Zou, J. 2025. Optimizing generative AI by backpropagating language model feedback. *Nature*, 639(8055): 609–616.

Zhang, L.; Zhao, T.; Ying, H.; Ma, Y.; and Lee, K. 2024a. OmAgent: A Multi-modal Agent Framework for Complex Video Understanding with Task Divide-and-Conquer. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 10031–10045. Miami, Florida, USA: Association for Computational Linguistics.

Zhang, W.; Tang, K.; Wu, H.; Wang, M.; Shen, Y.; Hou, G.; Tan, Z.; Li, P.; Zhuang, Y.; and Lu, W. 2024b. Agent-Pro: Learning to Evolve via Policy-Level Reflection and Optimization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5348–5375.

Zhang, Y.; Sun, P.; Jiang, Y.; Yu, D.; Weng, F.; Yuan, Z.; Luo, P.; Liu, W.; and Wang, X. 2022. Bytetrack: Multi-object tracking by associating every detection box. In *European conference on computer vision*, 1–21. Springer.

Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; and Chen, J. 2024. Detrs beat yolos on real-time object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 16965–16974.

Zhao, Y.; Misra, I.; Krähenbühl, P.; and Girdhar, R. 2023. Learning video representations from large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6586–6597.